Description:

Part 0 – Reading data:

* Datasets are read and imported in this part, which are adopted from movielens datasets (address: https://grouplens.org/datasets/movielens/).
* Also, with the help of the function …, we create another csv file (stars or actors list) via web scarping (from IMDb links).

Part 1 – Trimming data, feature (attribute) extraction and preliminary analysis:

* In this part, we trim and clean the datasets in order to make them compatible with each other.
* Another major task is to extract features out of the datasets – features such as movie genres, release year and stars list.
* We do a little of analysis through word-clouding, etc., to see which genre or year is more popular.
* We also define the similarity metric, which is (adjusted) cosine similarity criterion (adjusted cosine similarity is the same as the cosine one, but here average rating of a user is deducted from ratings in order to balance users’ ratings). To our understanding, This criterion seems to be working more realistically (in the movie recommender systems) compared to other criteria, such as Euclidean, Hamming or Jaccard distance.

Part 2 – Item-based recommender system:

* Based on (cosine) similarities between items (here, movies), we recommend those movies which a given user has not seen yet (or equivalently has not rated yet).
* We predict the ratings of those non-rated movies by using the weighted averaging method.

Part 3 – User-based recommender system:

* Based on (cosine) similarities between users, we recommend those movies which a given user has not seen yet (or equivalently has not rated yet).
* We predict the ratings of those non-rated movies by using the weighted averaging method.

Part 4 – Content-based (item-based collaborative filtering) recommender system:

* It’s the fun part of this mini-project. After extracting the features, we exploit them and analyze the users’ similarities with respect the feature in order to recommend (non-rated) movies to a specific user.

Enhancements:

There are several enhancements which can be applied to the recommender systems:

* Parallelization in order to read, write and process different tasks in parallel. This can be done in different spots, e.g., in the web scraping, recommendation to different users at a time, etc.
* Similarity metric can be more adjusted and customized. For example, when do we say that two users or items are similar? Should the users or items have at least a couple of things in common for us to say that they are similar.
* Use of binary ratings to make the program time- and memory-efficient.
* Recommendations can be done via machine learning algorithms (regression and neural networks models) and other types of decomposition methods (e.g., SVD, UV).
* Use of recommender system packages in R, such as recommenderlab, can be explored.
* This project is ongoing, so please reach to me via [amirpasha.shirazinia@gmail.com](mailto:amirpasha.shirazinia@gmail.com) …

Acknowledgment: I’d like to thank several humble guys who posted their work and codes online on blogs, webpages, etc., and inspired this mini-project. Just to name a few:

<https://muffynomster.wordpress.com/2015/06/07/building-a-movie-recommendation-engine-with-r/>

<https://www.r-bloggers.com/item-based-collaborative-filtering-recommender-systems-in-r/>